# Cómo Hacer un Diagrama de Dispersión

Hasta el momento, hemos dibujado solamente diagramas de barra con datos simples — solo conjuntos de datos uni-dimensionales.

Cuando se tienen dos conjuntos de datos y se quiere generar una gráfica en la cual se presentan los dos, es necesario tener una segunda dimensión. El diagrama de dispersión es una visualización común con la que se pueden representar dos conjuntos de datos relacionadas en dos ejes: horizontal y vertical, x y y.

## Los Datos

Como se pudo ver en *Tipos de Datos*, existe mucha flexibilidad en cuanto se refiere a la estructuración de los conjuntos de datos. Para el diagrama de dispersión, se va a utilizar un arreglo de arreglos. El arreglo primario contiene un elemento por cada “dato”. Cada uno de los elementos de este “dato” será a su vez otro arreglo, con dos datos, uno con el

valor de x y otro con el valor de y.

```
var dataset = [
            [5, 20], [480, 90], [250, 50], [100, 33], [330, 95],
            [410, 12], [475, 44], [25, 67], [85, 21], [220, 88]
          ];
```

Recuerde que [] significa arreglo, por consiguiente cuando se anidan los corchetes [[]] esto significa que existe un arreglo dentro de otro arreglo. Los elementos dentro de un arreglo se separan con comas, entonces un arreglo que contiene tres arreglos se muestra de la siguiente manera: [[].[].[]]

Nuestros datos se pueden volver a escribir de la siguiente manera, que es más fácil de leer:

```
var dataset = [
            [ 5,     20 ],
            [ 480,   90 ],
            [ 250,   50 ],
            [ 100,   33 ],
            [ 330,   95 ],
            [ 410,   12 ],
            [ 475,   44 ],
```

```
            [ 25,    67 ],
            [ 85,    21 ],
            [ 220,   88 ]
        ];
```

Ahora puede ver cada una de las diez filas le corresponderá un punto de nuestra visualización. En la fila [5,20] usaremos `5` como el valor de x y `20`para el valor de y.

## El Diagrama de Dispersión

Podemos traspasar la mayoría del código de nuestros experimentos con diagramas de barras, incluyendo el que genera el elemento SVG.

```
//Crear un elemento SVG
var svg = d3.select("body")
        .append("svg")
        .attr("width", w)
        .attr("height", h);
```

En vez crear `rect` (rectángulos) en este caso se crearán `circle` (círculos) para cada dato.

```
svg.selectAll("circle")
```

```
    .data(dataset)
    .enter()
    .append("circle")
```

Igualmente, en vez de especificar los atributos `x, y, width y height` de `rect`(rectángulo), nuestros `circle`(círculos) necesitan `cx, cy, y r`:

```
.attr("cx", function(d) {
    return d[0];
})
.attr("cy", function(d) {
    return d[1];
})
.attr("r", 5);
```

Acá está el diagrama de dispersión funcionando.

Tome nota cómo es que accedemos a los valores de los datos y los usamos para definir los valores de `cx` y `cy`. Cuando se usa la `funtion(d)`, D3 entrega automáticamente el valor actual de `d` a su función. En este caso el valor actual es uno de los arreglos pequeños dentro del arreglo más grande denominado `dataset`.

Cuando `d` es un arreglo de valores ( y no un solo valor, como 3.14159), es necesario usar la notación de corchetes para obtener estos valores. Por eso, en vez de `return d`(o devolver d), debemos usar `return d[0]` (devolver d[0]) y `return d[1]` (devolver d[1]), que devuelven los primeros dos valores del arreglo.

Por ejemplo, en el caso de nuestro primer data [5, 20], el primer valor (posición 0 del arreglo) es `5`, y el segundo valor (posición del arreglo 1) es `20`. Por consiguiente:

`d[0]` devuelve 5 `d[1]` devuelve 20

Por si acaso, si necesita obtener datos almacenados en un conjunto de datos grande (por fuera de D3), lo puede hacer usando la notación de corchetes. Por ejemplo.

`dataset[5] returns [410, 12]`

No me cree? Mire nuevamente el diagrama de dispersión, abra la consola de JavaScript y escriba `dataset[5] o dataset[5][1]` y mire qué pasa.

## Tamaño

De pronto quiere que sus círculos tengan diferentes tamaños, de tal manera que sus radios se definan en función de los valores de y. En vez de asignarle a todos los radios `r`, el valor de `5`, intente:

```
.attr("r", function(d) {
return Math.sqrt(h - d[1]);
});
```

Esto no es ni atractivo ni útil, pero ilustra cómo se puede usar `d` con la notación de corchetes para referenciar valores y asignárselos a `r`.

## Etiquetas

Pongámosle etiquetas a nuestros puntos con elementos `text`(texto). Acá voy a modificar el código de nuestros experimentos con diagramas de barras, empezando por:

```
svg.selectAll("text")
```

```
    .data(dataset)
    .enter()
    .append("text")
```

Hasta el momento, el código busca los elementos `text` dentro del SVG (no existen aún), y luego añade un elemento `text` por cada dato. Acá se utiliza el método `text()` para especificar el contenido de cada elemento.

```
.text(function(d) {
    return d[0] + "," + d[1];
})
```

Esto se ver desordenado, pero tenga paciencia. Acá otra vez estamos usando `funtion(d)` para accesar los datos. Luego, dentro de la función, estamos usando `d[0] y d[1]` para obtener los dos valores que están almacenados en el arreglo de datos.

Los signos de +, cuando se usan con las cadenas de caracteres, tal como la coma entre las comillas “,” actúan como operadores de *concatenación*. Entonces lo que está pasando en esta línea de código realmente es: Obtenga los valores `d[0] y d[1]`y únalos con una coma en la mitad. El resultado es algo similar a `5,20 o 25, 67`.

Ahora especificamos *donde* se va ubicar el texto, con los valores `x` y `y`. Por el momento,

usemos `d[0]` y `d[1]`, los mismos valores que usamos para especificar las posiciones del círculo (`circle`).

```
.attr("x", function(d) {
    return d[0];
})
.attr("y", function(d) {
    return d[1];
})
```

Por último, vamos a añadir algo de estilo con:

```
.attr("font-family", "sans-serif")
.attr("font-size", "11px")
.attr("fill", "red");
```

Acá está el código funcionando.

## Siguientes Pasos

Ojalá que los conceptos fundamentales de D3 ya estén quedando claros: cómo cargar datos, generar nuevos elementos, y usar valores de datos para generar los atributos de esos elementos.

Sin embargo, la imagen de arriba a duras penas pasa por una visualización de datos. El diagrama de dispersión es difícil de leer y el código no es muy flexible. Honestamente, aún no hemos mejorado – broma – con respecto al *Asistente de Diagramas* de Excel!

No hay por qué preocuparse: D3 es mucho más interesante que el Asistente de Gráficos (y eso sin mencionar Clippy), pero para generar un diagrama interactivo y atractivo requiere que subamos de nivel en nuestra destreza de D3. Para usar datos en forma flexible, vamos a aprender acerca de las *escalas* de D3. Para que nuestro diagrama de dispersión sea más fácil de leer, vamos a aprender acerca de los *generadores de ejes* y las etiquetas de los ejes. Luego, vamos a hacer que las cosas sean interactivas y vamos a aprender cómo actualizar los datos sobre la marcha.

Siguiente: Escalas –>

---

Este tutorial fue traducido por Gabriel Coch
Todo el contenido fue desarrollado por y le pertenece a Scott Murray